UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/656,097	09/05/2003	Bhushan Khaladkar	OI7035732001	9920

23639 7590 03/08/2006

BINGHAM, MCCUTCHEN LLP
THREE EMBARCADERO CENTER
18 FLOOR
SAN FRANCISCO, CA 94111-4067

EXAMINER	
TSUI, WILSON W	
ART UNIT	**PAPER NUMBER**
2178	

DATE MAILED: 03/08/2006

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	10/656,097	KHALADKAR ET AL.	
	Examiner	Art Unit	
	Wilson Tsui	2178	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *05 September 2003*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-35* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-35* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. This action is in response to application filed on September 5, 2003.

2. Claims 1-35 are pending. Claim 1, 11, 18, 30, 31, 32, 33, and 35 are independent claims.

Claim Rejections - 35 USC § 102

The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

3. Claims 18, 19, 20, 23, 24, 29, 34, and 35 are rejected under 35 U.S.C. 102(e) as being anticipated by Marcy (US Patent: 6,662,342 B1, published: Dec. 9, 2003, filed: Dec. 13, 1999).

With regards to claim 18, Marcy teaches a method comprising:

- *Receiving the schema for the data that is based on the mark-up language* (column 4, lines 40-54: whereas, a XML parser receives a schema of XML data (DTD) for processing)
- *Identifying a child node that is to be accessed within the data*: An application program is used to access content in an instance of XML data (column 6, lines 23-32: whereas, "the application program can obtain information and content on any part of the document"). Furthermore, it is inherent that the application

program identifies a child node to be accessed within the data since the application access any part of the document, which includes child nodes of elements (column 5, lines 57-59).

- *Reviewing the schema to determine one or more access parameters relating to the child node*: A schema of XML data (DTD) is reviewed, and handles are created (column 4, lines 40-55:). These handles are used to reference nodes for each element and/or attribute for direct access, since metadata for each node includes memory location information (column 6, lines 2-11).
- *Using the one or more access parameters to directly access the child node without requiring progressive traversal of child nodes* (column 6, lines 23-32: whereas, an application program uses location information for a specified node (such as a child node), to directly reference the corresponding data in an XML instance).

With regards to claim 19, Marcy teaches a method *in which the mark-up language is based on XML* (column 4, lines 40-54: whereas, XML markup language used).

With regards to claim 20, Marcy teaches a method *in which the access parameters are based on offset position* (column 6, lines 60-65: whereas, memory offset locations are used as access parameters).

With regards to claim 23, which depends on 18, Marcy teaches a method in which *the child node is not directly accessed if the node is not defined in the schema* (Fig. 3, column 5, lines 1-36: whereas, all elements, including child nodes that are not defined in the schema do not get referenced with an access handle. Since the user

application (reference 6) does not receive the handle, the child node cannot be referenced/directly accessed).

With regards to claim 24, which depends on claim 18, Marcy teaches a method comprising *directly accessing a child node in which the method is performed* (in claim 18, and is rejected under the same rationale). Furthermore, Marcy teaches implementing the method using Java (column 8, lines 11-17), and thus it is inherent that a method implemented in Java will *support the system's native datatypes* using the system's Java Virtual Machine.

With regards to claim 29, which depends on claim 18, Marcy teaches a method *in which other child nodes not presently needed are not loaded into memory* (column 7, lines 10-20: whereas, the child node data/values are not loaded into memory, since the application program references a memory location in XML instance.)

With regards to claim 34, for a system performing a method similar to claim 18, and is rejected under the same rationale.

With regards to claim 35, for a computer program product comprising a computer usable medium having executable code, performing a method similar to claim 18, and is rejected under the same rationale.

Claim Rejections - 35 USC § 103

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the

invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

4. Claims 1, 2, 3, 4, 8, 10, 11, 12, 14, 16, 21, 30, 31, 32, and 33 are rejected under 35 U.S.C. 103(a) as being unpatentable over Marcy (US Patent: 6,662,342 B1, published: Dec. 9, 2003, filed: Dec. 13, 1999) in view of Dreyband et al (US Patent Application: US 2001/0029604 A1, published: Oct. 11, 2001, filed: Apr. 27, 2001).

With regards to claim 1, Marcy teaches a method comprising:

- *Receiving a schema for the XML data*, similarly in claim 11, and is rejected under the same rationale.
- Using an access procedure for *providing direct access to the element within an instance of the XML data, without requiring progressive traversal of child nodes*, similarly in claim 11, and is rejected under the same rationale.

However, Marcy does not teach a system comprising *implementing a named access procedure, and creating the named access procedure and associating the named access procedure with the element when the element is appropriate for association.*

Dreyband et al teaches a system comprising:

- *Identifying an element within the schema to associate with a named access procedure* (Figure 3, paragraph 0029: whereas, elements defined in the xml file are identified, such as the element used in Dreyband's example xml file labeled: "name")
- *Determining if the element identified is appropriate for association with a named access procedure* (paragraph 0028: whereas, a check is performed to make sure

that the xml document being used is "schema valid", thus inherently, the element defined in the xml document has been identified to be appropriate also).

- *Creating the named access procedure and associating the named access procedure with the element when the element is appropriate for association* (Figure 3, paragraph 0029: whereas, using the schema, an element is associated with a named access procedure by creating a java class named: "person", and having a named access procedure/attribute labeled "getName" (getName is associated with the element labeled "name" in the provided schema shown in Figure2)).

It would have been obvious to one of the ordinary skill in the art at the time of the invention to have modified Marcy's access procedure for directly accessing an instance of XML data to further include the implementation of a named access procedure system as taught by Dreyband et al. The combination of Marcy and Dreyband et al would have allowed Marcy's system to be able to access XML data through a defined named access approach using a schema as a standard for association.

With regards to claim 2, which depends on claim 1, for a method performing a similar method to claim 12, is rejected under the same rationale.

With regards to claim 3, which depends on claim 2, Marcy teaches a method comprising:

- *An access procedure:* (column 6, lines 23-33: whereas, the application program makes the determination to access an element by using an access procedure

that includes referencing a memory location associated with the element for direct access).

- *Access parameters are based on offset position for each element*: Each element in a XML document (Figure 1) is assigned a memory offset location to be sent to an application for use as access parameters (column 6, lines 60-65) .

However, Marcy does not teach access parameters *are used to define the named access procedure*

Dreyband et al teaches the use of an object oriented implementation for associating xml elements with a *named access procedures/methods*, in claim 1, and is rejected under the same rationale.

It would have been obvious to one of the ordinary skill in the art at the time of the invention to have modified Marcy's method of using offsets for each element as access parameters, to be further defined in the named access procedure taught by Dreyband et al. The combination of Marcy and Dreyband et al would have allowed Marcy's system to have been able to access XML data through a defined named access approach using a schema as a standard for association.

With regards to claim 4, which depends on claim 1, Marcy does not teach a method which *the named access procedure is a procedure to get a value for the element or to set a value for the element.*

However, Dreyband et al teaches a method *in which a named access procedure is defined to get a value for the child node or to set a value for the child node* (Figure 2 and 3: whereas, child elements disclosed in a schema shown in Figure 2 (such as child

node 'Name'), have corresponding get and set functions shown in Figure 3 (such as 'getName' and 'setName')).

It would have been obvious to one of the ordinary skill in the art at the time of the invention to have modified the access procedure used by Marcy's application program to use a named access procedure to get and/or set a child node value as taught by Dreyband et al. The combination of Marcy and Dreyband et al would have allowed Marcy to have implemented a mapping of a XML schema into Java with "bean style" mutator and accessor methods (paragraph 0029).

With regards to claim 8, which is dependent on claim 1, Marcy does not teach *an element is not appropriate for association if it is a node not defined in the schema.*

However, Dreyband et al teaches *an element is not appropriate for association if it is a node that is not defined in the schema* (paragraph 0044: whereas, Dreyband et al's system checks if the user xml document is schema valid). Dreyband et al's system only makes the association if the user's document is schema valid (paragraph 0029: whereas, a mapping happens when the user's document is schema valid).

It would have been obvious to one of the ordinary skill in the art at the time of the invention to have modified Marcy's system to further include associating elements with named access procedures only on the basis of a valid schema as taught by Dreyband et al. The combination of Marcy and Dreyband et al would have allowed Marcy's system to have been able to incorporate an error checking procedure when performing an association.

With regards to claim 10, which is dependent on claim 1, Marcy does not teach *the named access procedure is implemented as a bean accessor type.*

However, Dreyband et al teaches *the named access procedure* (in claim 1, and is rejected under the same rationale), *is implemented as a bean accessor type* (Figure 2 and 3: whereas, child elements disclosed in a schema shown in Figure 2 (such as child node 'Name'), have corresponding get and set functions shown in Figure 3 (such as 'setName')).

It would have been obvious to one of the ordinary skill in the art at the time of the invention to have modified Marcy's system to include the named access procedure with a bean accessor type as taught by Dreyband et al. The combination of Marcy and Dreyband et al would have allowed Marcy's system to directly access an XML instance using a common accessor interface.

With regards to claim 11, Marcy teaches a method comprising:

- *Identifying an element within an instance of the XML data to access* (column 6, lines 23-33: whereas, an application program issues a request to access an element within an instance of XML data).
- *Determining if the element has been associated with an* access procedure *corresponding to the element,* an access procedure *providing direct access to the element within an instance of the XML data without requiring progressive traversal of child nodes, and if the element has been associated with an access procedure, then using* an access procedure *to access the element in the instance of the XML data* (column 6, lines 23-33: whereas, the application program makes

the determination to access an element by using an access procedure that includes referencing a memory location associated with the element for direct access).

- *If the element has not been associated with the named access procedure, then using a DOM API to access the element in the instance of the XML data* (column 2, lines 1-22: whereas, a DOM API is used to access element data through a search procedure)

However, Marcy does not teach implementing a *named access procedure that is associated with an element.*

Dreyband et al teaches a method for *implementing named access procedure that is associated with an element*; similarly in claim 1, and is rejected under the same rationale.

It would have been obvious to one of the ordinary skill in the art at the time of the invention to have modified Marcy's access procedure to use the named access procedure approach taught by Dreyband et al. The combination of Marcy and Dreyband et al, would have allowed Marcy's system to have "mapped the entire schema into the object oriented language" (paragraph 0011).

With regards to claim 12, which depends on claim 11, Marcy teaches *a schema for XML data is known apriori* (column 4, lines 40-54: whereas, a XML parser receives a schema of XML data (DTD) for processing. However, Marcy does not explicitly teach *the named access procedure is based upon analysis of the schema.*

Dreyband et al teaches a method *the named access procedure is based upon analysis of the schema* (Dreyband et al, paragraph 0029: whereas, a named access procedure/method is defined based on a valid schema, and the respective element defined in it).

It would have been obvious to one of the ordinary skill in the art at the time of the invention to have modified Marcy's method for schema analysis to include Dreyband et al's method for implementing a named access procedure based upon the schema analysis. The combination of Marcy and Dreyband et al would have allowed an application to have used Marcy's system based upon a predefined method of information exchange.

With regards to claim 14, which depends on claim 11, Marcy teaches a method *in which other elements of the data not presently needed are not loaded into memory* (column 7, lines 10-20: whereas, values at child nodes are not loaded into memory, since the application program directly references a memory location in XML).

With regards to claim 21, which depends on claim 18, Marcy does not explicitly teach method *in which a named access procedure is defined to get a value for the child node or to set a value for the child node.*

However, Dreyband et al teaches a method *in which a named access procedure is defined to get a value for the child node or to set a value for the child node* (Figure 2 and 3: whereas, child elements disclosed in a schema shown in Figure 2 (such as child node 'Name'), have corresponding get and set functions shown in Figure 3 (such as 'getName' and 'setName')).

It would have been obvious to one of the ordinary skill in the art at the time of the invention to have modified the access procedure used by Marcy's application program to use a named access procedure to get and/or set a child node value as taught by Dreyband et al. The combination of Marcy and Dreyband et al would have allowed Marcy to have implemented a mapping of a XML schema into Java with "bean style" mutator and accessor methods (paragraph 0029).

With regards to claim 30, Marcy teaches a system for:

- *Means for receiving a schema for the XML data*, similarly in claim 11, and is rejected under the same rationale.
- *Means for* using an access procedure for *providing direct access to the element within an instance of the XML data, without requiring progressive traversal of child nodes*, similarly in claim 11, and is rejected under the same rationale.

However, Marcy does not teach a system comprising *means for implementing a named access procedure, and means for creating the named access procedure and associating the named access procedure with the element when the element is appropriate for association.*

Dreyband et al teaches a system comprising:

- *Means for identifying an element within the schema to associate with a named access procedure*, similarly in claim 1, and is rejected under the same rationale.
- *Means for determining if the element identified is appropriate for association with a named access procedure*, similarly in claim 1, and is rejected under the same rationale.

- *Means for creating the named access procedure and associating the named access procedure with the element when the element is appropriate for association*, similarly in claim 1, and is rejected under the same rationale.

It would have been obvious to one of the ordinary skill in the art at the time of the invention to have modified Marcy's access procedure for directly accessing an instance of XML data to further include the implementation of a named access procedure system as taught by Dreyband et al. The combination of Marcy and Dreyband et al would have allowed Marcy's system to have been able to access XML data through a defined named access approach using a schema as a standard for association.

With regards to claim 31 for a computer program product comprising a computer usable medium having executable code, is similar to a system performing a similar method to claim 30, and is rejected under the same rationale.

With regards to claim 32, for a system performing a method similar to claim 11, and is rejected under the same rationale.

With regards to claim 33, for a computer program product comprising a computer usable medium having executable code, is similar to a system performing a method of claim 11, and is rejected under the same rationale.

5. Claims 15, 16, and 17 are rejected under 35 U.S.C. 103(a) as being unpatentable over Marcy (US Patent: 6,662,342 B1, published: Dec. 9, 2003, filed: Dec. 13, 1999) and Dreyband et al (US Patent Application: US 2001/0029604 A1, published: Oct. 11, 2001, filed: Apr. 27, 2001) in view of Wan (US Patent: 2003/0233618 A1, published: Dec. 18, 2003, filed: Jun. 16, 2003, Foreign priority: Jun. 17, 2002).

With regards to claim 15, which depends on claim 11, Marcy does not teach a method in which *the element is at a known offset from a parent location.*

However, Wan teaches *the element is at a known offset from a parent location* (Table B and D: whereas, Table B represents an XML instance (which includes child nodes), while Table D shows a method for accessing a memory location of the child nodes using offsets based off of the memory location of a parent node).

It would have been obvious to one of the ordinary skill in the art at the time of the invention to have modified Marcy's application for accessing node data to further include the method for accessing child nodes via offsets based off of a location of the corresponding parent nodes as taught by Wan. The combination of Marcy, Dreyband et al and Wan would have allowed Marcy's system to have been able to have reduced iterative traversal of an XML instance.

With regards to claim 16, which depends on claim 15, Marcy teaches a method *in which the known offset is managed independently of the XML data* (column 6, lies 60-65, Figure 3: whereas, an application program (reference number 6), receives offset information from the XML parser, and thus, is managed independently of the data).

With regards to claim 17, which depends on claim 11, Marcy does not teach a method *in which a memory layout associated with the XML data is maintained as a flat layout.*

However, Wan teaches a method *in which a memory layout associated with the XML data is maintained as a flat layout* (Table B and D: whereas, Table B represents an

XML instance (which includes child nodes), while Table D shows a method for accessing a memory location of the child nodes in a flat layout (index/list)).

It would have been obvious to one of the ordinary skill in the art at the time of the invention to have modified Marcy's application for accessing node data to further include the method for accessing the memory location of nodes in a flat layout. The combination of Marcy and Wan would have allowed Marcy's system to have been able to improve the usage of memory.

6. Claims 25, 26, 28, and 27 are rejected under 35 U.S.C. 103(a) as being unpatentable over Marcy (US Patent: 6,662,342 B1, published: Dec. 9, 2003, filed: Dec. 13, 1999) in view of Wan (US Patent: 2003/0233618 A1, published: Dec. 18, 2003, filed: Jun. 16, 2003, Foreign priority: Jun. 17, 2002).

With regards to claim 25, which depends on claim 18, Marcy does not teach a method *in which direct access is performed to an offset location for the child node.*

However, Wan teaches a method *in which direct access is performed to an offset location for the child node* (Table B and D: whereas, Table B represents an XML instance (which includes child nodes), while Table D shows a method for accessing a memory location of the child nodes using offsets).

It would have been obvious to one of the ordinary skill in the art at the time of the invention to have modified Marcy's application for accessing node data, to further include the method for accessing child nodes via offsets as taught by Wan. The combination of Marcy and Wan would have allowed Marcy system to have parsed

"schemas of structural documents to determine predefined deterministic relationships between components of structured documents to be indexed" (paragraph 0009).

With regards to claim 26, which depends on claim 25, Marcy does not teach a method *in which the child node is at a known offset from a location for the parent node.*

However, Wan teaches a method *in which the child node is at a known offset from a location for the parent node* (Table B and D: whereas, Table B represents an XML instance (which includes child nodes), while Table D shows a method for accessing a memory location of the child nodes using offsets based off of the memory location of a parent node).

It would have been obvious to one of the ordinary skill in the art at the time of the invention to have modified Marcy's application for accessing node data to further include the method for accessing child nodes via offsets based off of a location of the corresponding parent nodes as taught by Wan. The combination of Marcy and Wan would have allowed Marcy's system to have been able to have reduced iterative traversal of an XML instance.

With regards to claim 27, which depends on claim 26, Marcy teaches a method *in which a mapping of the known offset is managed independently of the data* (column 6, lies 60-65, Figure 3: whereas, an application program (reference number 6), receives offset information from the XML parser, and thus, is managed independently of the data).

With regards to claim 28, which depends on claim 25, Marcy does not teach a method *in which memory layout associated with the data is maintained as a flat layout.*

However, Wan teaches a method *in which memory layout associated with the data is maintained as a flat layout* (Table B and D: whereas, Table B represents an XML instance (which includes child nodes), while Table D shows a method for accessing a memory location of the child nodes in a flat layout (index/list)).

It would have been obvious to one of the ordinary skill in the art at the time of the invention to have modified Marcy's application for accessing node data to further include the method for accessing the memory location of nodes in a flat layout. The combination of Marcy and Wan would have allowed Marcy's system to have been able to improve the usage of memory.

7. Claims 5, 6, 7, 9, and 13 are rejected under 35 U.S.C. 103(a) as being unpatentable over Marcy (US Patent: 6,662,342 B1, published: Dec. 9, 2003, filed: Dec. 13, 1999) and Dreyband et al (US Patent Application: US 2001/0029604 A1, published: Oct. 11, 2001, filed: Apr. 27, 2001) in view of JAXB (Sun Microsystems, pages 58, and 74, published: January 8, 2003).

With regards to claim 5, which depends on claim 1, Marcy does not explicitly teach a method *in which direct mapping is performed to an intended datatype for the element.*

However, JAXB teaches a method *in which direct mapping is performed to an intended datatype for an element* (page 74: whereas, based on the datatype disclosed in a schema, a mapping is performed to an intended Java datatype, as shown in the example, an 'int' datatype in the schema is mapped to an 'int' Java datatype, and a 'float' datatype in the schema is mapped to a 'float' Java datatype).

It would have been obvious to one of the ordinary skill in the art at the time of the invention to have modified Marcy's system to include a method of directly mapping datatypes as taught by JAXB. The combination of Marcy and JAXB would have allowed an application that utilized Marcy's system, to have accessed data more efficiently, without the overhead of any additional datatype mapping steps.

With regards to claim 6, which depends on claim 5, Marcy does not explicitly teach a method *in which a conversion to a string datatype is not performed when mapping to the intended datatype.*

However, JAXB teaches a method in *which a conversion to a string datatype is not performed when mapping to the intended datatype* (page 74: whereas, an 'int' mapping is performed).

With regards to claim 7, which depends on claim 5, Marcy does not explicitly teach a method *in which the mapping is a close-matching datatype.*

However, JAXB teaches a method in *which the mapping is a close matching datatype* (page 58: whereas, mapping to a javatype is based on the defined schema datatype. As shown, an 'unsigned int' datatype in a schema is mapped to the closest matching datatype in Java, which is a 'long').

It would have been obvious to one of the ordinary skill in the art at the time of the invention to have modified Marcy's XML access system to further provide a closest matching datatype that is supported for a particular system/application. The combination of Marcy and JAXB would have allowed Marcy's system operate flexibly on various different platforms using different datatypes.

With regards to claim 9, which depends on claim 1, Marcy does not explicitly teach a method *in which the element is appropriate for association if it corresponds to a native datatype of the system in which the method is performed* (page 58, Table 5-1: whereas, all the schema datatypes disclosed are mapped to all supported Java datatypes. Furthermore, it is inherent that Java datatypes will be supported by the native datatypes available in the system through the use of the platform specific Java Virtual Machine).

It would have been obvious to one of the ordinary skill in the art at the time of the invention to have modified Marcy's system to further include a method for associating elements that map to Java datatypes for system independent datatype support. The combination of Marcy and JAXB would have allowed Marcy's system to have operated independent of platform type.

With regards to claim 13, which depends on claim 11, for a method performing a similar method to claim 5, is rejected under the same rationale.

8. Claim 22 is rejected under 35 U.S.C. 103(a) as being unpatentable over Marcy (US Patent: 6,662,342 B1, published: Dec. 9, 2003, filed: Dec. 13, 1999) in view of JAXB (Sun Microsystems, pages 58 and 74, published: January 8, 2003).

With regards to claim 22, which depends on claim 18, Marcy teaches a method comprising *identifying a child node* that is to be accessed within the data: An application program is used to access content in an instance of XML data (column 6, lines 23-32: whereas, "the application program can obtain information and content on any part of the document"). Furthermore, it is inherent that the application program identifies a child

node to be accessed within the data since the application access any part of the document, which includes child nodes of elements (column 5, lines 57-59). However, Marcy does not explicitly teach a method *in which direct mapping is performed to an intended datatype for the* child node.

JAXB teaches a method *in which direct mapping is performed to an intended datatype for a child node/*element (page 74: whereas, based on the datatype disclosed in a schema, a mapping is performed to an intended Java datatype, as shown in the example, an 'int' datatype in the schema is mapped to an 'int' Java datatype, and a 'float' datatype in the schema is mapped to a 'float' Java datatype).

It would have been obvious to one of the ordinary skill in the art at the time of the invention to have modified Marcy's method of accessing child node data to further include a method of direct mapping to an intended datatype as taught by JAXB. The combination Marcy and JAXB would have allowed Marcy's system to have performed faster child node data retrieval.

Conclusion

The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

- Fry et al (US Application: US 2003/0163603, published: Aug. 28, 2003, filed: Nov. 26, 2002): Teaches XML data binding and schema analysis for validating and non-validating parsers.
- Tian et al (SIGMOD, published: March 2002, pages 5-10): Teaches the use of offsets for memory access and a flat memory structure.

- Call (US Application: US 2002/0143521 A1, published: Oct. 3, 2002, filed: Dec. 10, 2001): Teaches an improved method for XML parsing without using the DOM tree representation.
- Golden (US Application: US 2002/ 0073399, published: Jun. 13, 2002, filed: Dec. 8, 2000): Teaches databinding, data mapping , JavaBean implementation.
- Noga et al (ACM, published: November 2002, pages 88-94): Teaches lazy XML processing to optimize memory usage.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Wilson Tsui whose telephone number is (571)272-7596. The examiner can normally be reached on Monday - Friday.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Stephen Hong can be reached on (571) 272-4124. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

W.T. March 3, 2006

Wilson Tsui
Patent Examiner
Art Unit: 2178
March 3, 2006

STEPHEN HONG
SUPERVISORY PATENT EXAMINER